Variable Data Printing

PackEdge Workflow Manual

ESKO

Contents

1. Introduction

Variable Data Printing provides you the ability to print jobs with variable elements on digital presses while keeping in mind the needs of the packaging industry. A variable element can either be text, a barcode or an image.

This document will explain only those topics related to Variable Data Printing. Knowledge of Esko BackStage Pilot, Esko FlexRip, Esko PackEdge, Esko Plato and SmartMarks is required.

1.1 Copyright Notice

Correspondence regarding this publication should be forwarded to:

Esko Software BVBA

Kortrijksesteenweg 1095

B – 9051 Gent

info.eur@esko.com

2. Components

The Variable Data Printing module consists of a front end, a database and some BackStage server tasks.

2.1 Front End

The front end to the system consists of Esko PackEdge for creating a design, and Esko Plato for setting up the step and repeat. SmartMarks are used in PackEdge to define variable elements within the design.

2.1.1 Introducing SmartMarks

SmartMarks are objects in the design that are linked to the database entries, and are generated based on their contents. In other words, for various database records various SmartMark objects are generated.

A SmartMark can be of three different types – a Text Mark, a Barcode Mark or an Image Mark:

- A Text Mark contains a text value and implements tools for its formatting;
- A Barcode Mark carries a bar code; and
- An Image Mark is used to place an image into the design, based on its file location.

2.1.2 Plato for Step and Repeat

Esko Plato is used to set up the step and repeat parameters.

In Plato, you can specify general repetition parameters, such as plate and sheet size, horizontal and vertical steps, as well as variable-data-specific parameters, such as the database parameters, ordering pattern, and gaps generation.

2.1.3 Front End Output

The output of the front end consists of a repetition file (PDFPLA file) and a one-up file (Normalized PDF file).

Server tasks are started via the BackStage Pilot application. A more detailed description of the front end and its features can be found in the chapter on the PackEdge Workflow.

2.2 Database

CSV and XML database formats are supported. Both formats must be stored in UTF-8 encoding. CSV and XML files can be aquired from almost any database application, and as a consequence, other proprietary file formats are not supported.

2.2.1 Field name requirements

When composing database field names you are restricted to alphanumerical characters or underscores. The first character of a field name should not be a number. The interpretation of field names is case insensitive. Uppercase letters will always be converted to lowercase during import.

2.2.2 XML document structure requirements

The XML document structure should look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<rootElem>
    <recordElem>
        <fieldName1>FieldValue-1-1</fieldName1>
        <_fieldName2> FieldValue "1-2"</fieldName2>
    </recordElem>
    <recordElem>
        <fieldName1></fieldName1>
        <_fieldName2>FieldValue-2-2</fieldName2>
    </recordElem>
</rootElem>
```

The names of the XML elements are arbitrary, but the following rules must be observed:

- Field name elements must conform to the rules for valid field names listed above.
- The amount and type of field name elements must be the same for each record.

Note:

The use of element attributes in the XML database is not forbidden, but they are not interpreted by our solution in any way.

A correct XML file can easily be exported from Microsoft Access. Microsoft Excel XML files are not supported.

2.2.3 CSV format requirements

An example of the CSV format for the same database is provided below:

```
fieldName1,_fieldname2
FieldValue-1-1,"FieldValue ""1-2"""
,FieldValue-2-2
```

Aside from the standard rules creating CSV formatted files, the following must be taken into account:

- Field names must conform to the rules for valid field names listed above.
- Each record (line) must have the same number of fields, equal to the number of field names.
- Newline characters in field values are not supported (if you want to use a newline character in Text Mark, use the standard '[nl]' control sequence).

Field names can be extracted either from the first record of the CSV file, or can be provided in a separate CSV file. The delimiter character can be a comma, semicolon, space or a tab character, with the comma being the default.

A valid CSV file can easily be exported from either Microsoft Access or from Microsoft Excel.

Note:

If a database contains regional characters, files exported from Microsoft Excel have to be converted to **UTF-8 encoding** (e.g. using Notepad), as Microsoft Excel does not support UTF-8 encoding for CSV files.

Microsoft Excel for Mac is not able to export some of the regional characters into CSV format, which might lead to loss of data if it is used for generating the CSV database file.

2.3 BackStage Server Tasks

The BackStage Server offers two tasks to handle variable data in a PackEdge workflow: Expand Variable Data to PPML File, and RIP To WS4000 Series.

The Expand Variable Data to PPML File, and RIP To WS4000 Series can be chained into a workflow ticket.

2.3.1 The Expand Variable Data to PPML File task

The Expand Variable Data to PPML File task performs expansion of the specified database entries into the corresponding graphical output.

The task takes the following input files:

- a design file containing variable elements generated by Esko PackEdge (Normalized PDF file);
- a repetition file generated by Esko Plato (PDFPLA file);
- and a database file (XML or CSV).

It generates the following output files:

- one file with constant (unchanging) graphic elements (Normalized PDF file)
- several files with variable graphic elements (Normalized PDF files).

All these files are linked together by a standard PPML file.

2.3.2 The Rip to WS4000 Series task

The RIP to WS4000 Series task triggers FlexRip to create HP Indigo proprietary JLT files.These files can be read only by an HP Indigo press. The main difference compared to the generic RIP tasks is that this task supports PPML files, which is essential for correct variable data interpretation.

3. PackEdge Workflow

There are several ways to prepare documents and data for Variable Data Printing in Esko PackEdge and Esko Plato. There is, however a recommended workflow.

Use the following workflow for best results:

1. Create a job in Esko BackStage.
2. Prepare the data.
3. Design a one-up in Esko PackEdge.
4. Step and repeat in Esko Plato.
5. Launch server tasks in the BackStage Pilot.

3.1 Create a Job

The BackStage job will be used to encapsulate all the variable data resources.

To create a job:

- In the BackStage Pilot, create a new job or use an existing job.

  This job should contain a one-up document containing the design with variable elements definitions, and all the documents which are created during the workflow process (such as a repetition file, PDF files for both fixed and variable parts of the job and a linking PPML file). In addition, input data (database files and images) is also often stored in the Jobfolder.

3.2 Prepare the Data

An XML or CSV database file containg variable data values will be used for expansion. During the design of a one-up it is not needed to have access to the real (and often big) database. Instead, a small subset with just several records can be used. The real database will be needed only during the expansion process.

To prepare the data:

- Prepare a database file (in XML or CSV format).
- Make sure any variable images are stored in a BackStage Container.

  When variable images are used in the design, the files must be stored in a BackStage Container so that BackStage server can reach them. It is recommended to store all images in a Jobfolder. If the same set of images is used for multiple jobs it is possible to store them only once and reuse them from all jobs. All images must be in Esko native format (CT) and they cannot use RGB or LAB color space.

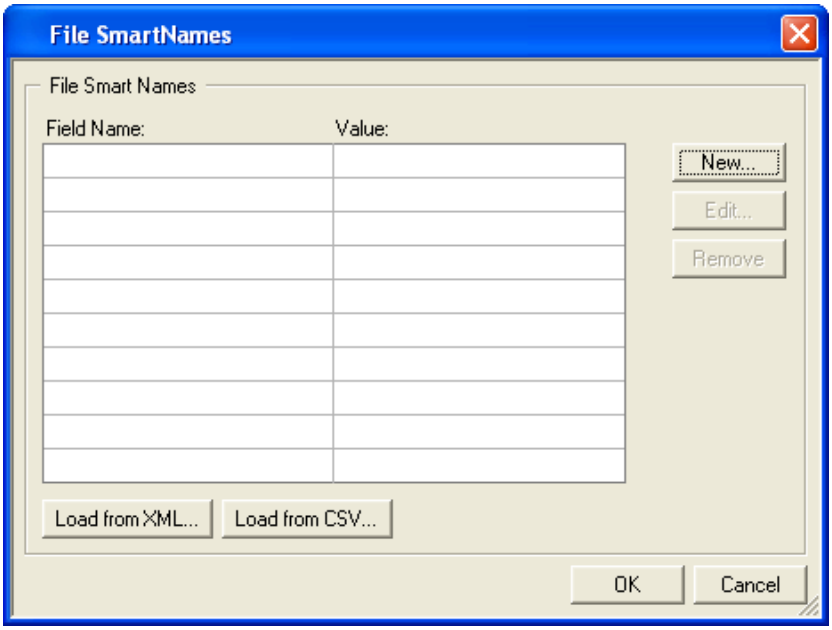
3.3 Design a one-up in Esko PackEdge

A design for Variable Data Printing consists of two main parts: static graphic elements in the background, and variable elements placed on top of them. Placing fixed graphics on top of the variable graphics is not supported.

To design a one-up:

1. Create a new file or open an existing file with a design proposal for the static part of the one-up.
2. Choose File SmartName from the Production menu to define File SmartNames.

   Designers do not need to have access to the actual database used for printing. However, they must be able to obtain the column names to use them in the variable part. This can be done either by using a dummy database file or by entering the field names and preview values manually.



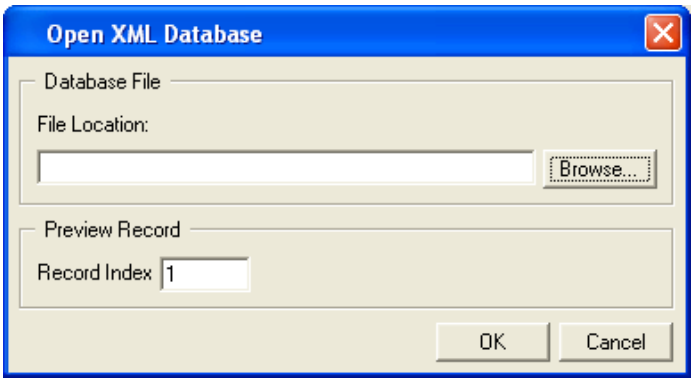
3. Do one of the following:

If you want to use a...	Then you must...
XML database	Click Load from XML… to open the database file. Proceed to step 4.
CSV database	Load from CSV… to open the database file. Proceed to step 5.

   It is also possible to manually define File SmartNames by clicking the 'New...' and 'Edit…' buttons. However, be careful not to misspell the column names, because during expansion only the entries in the database that are based on these names will be updated.

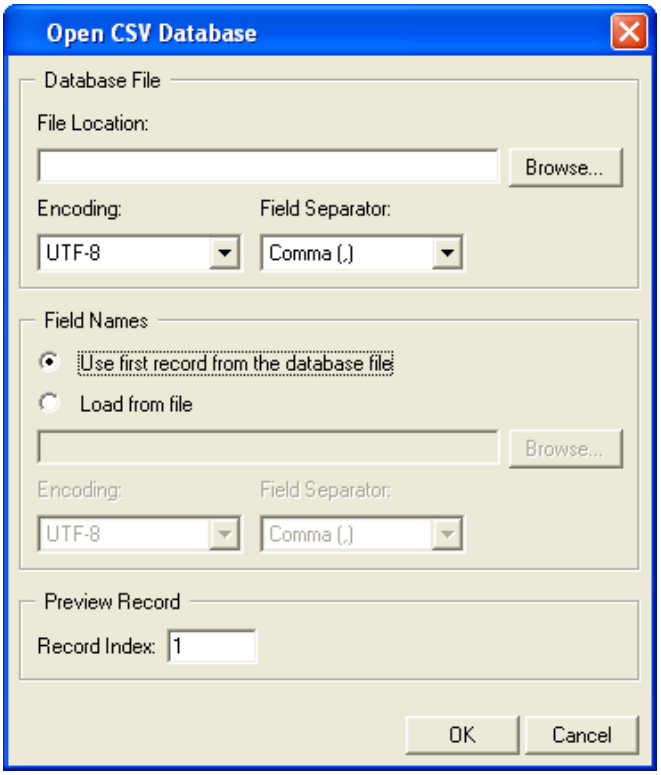
4. Click Browse in the Open XML Database dialog box and choose the XML file that contains the variable data.

'Record Index' is used to specify what record from the database will be used for the preview File SmartName values.



5. Click Browse in the Open CSV Database dialog box and choose the CSV file that contains the variable data. Then complete the additional settings for the CSV format.

   You have to specify the database encoding, a field separator character and a source for field names.

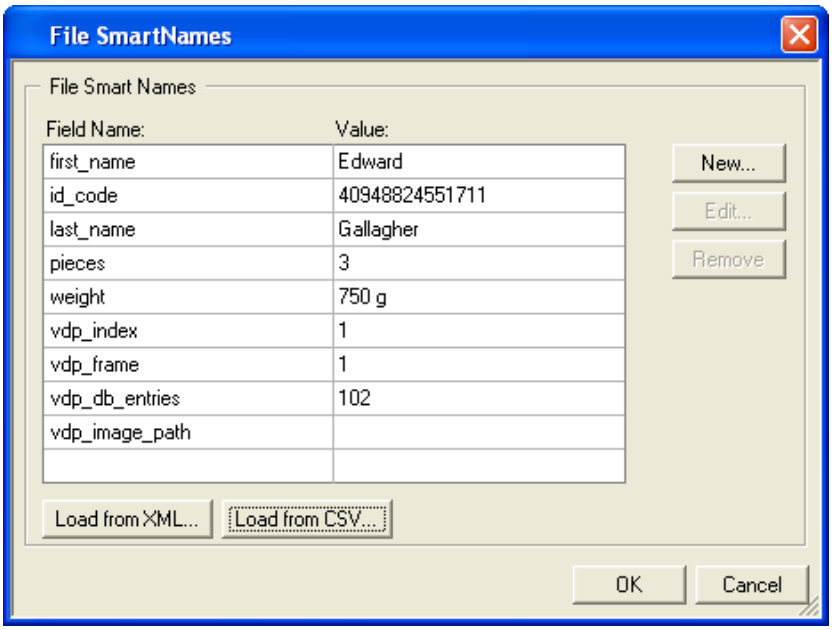


Currently only the UTF-8 encoding is supported for the database file. A field separator can be chosen from a predefined list, or can be set manually to a specific character.

Field names can be defined using the first record from a database. This will cover the majority of use cases. However, sometimes the field names may be located in a separate file. If so, you

will have to specify the path, encoding and a field separator for this file, because they can be different from the database file.

6. Click OK.
   The File SmartNames dialog box now displays the values loaded from a database.



Note:

The following SmartNames are generated automatically and you are strongly disadvised to change their field name:

- vdp_index: returns the number of the used record. During expansion of the very first record of the database its value is 1. For the second record it is 2 and so on.
- vdp_frame: returns the number of the current frame generated by the expansion task.
- vdp_db_entries: returns the total count of all entries in the current database file.
- vdp_image_path:

• vdp_image_path: you can use this SmartName to define the path of the folder containing the image files if you are working with Image Marks.

7. Click OK to submit the database.

You can now create or modify the variable data design. All variable objects are created as SmartMarks. Everything else will be stored in a fixed background.

Note:

Make sure that the design does not contain any fixed objects covering the VDP SmartMark objects, since all variable elements are placed on top of the fixed background. All SmartMark objects created in PackEdge will go into the variable layer during expansion. Even ordinary SmartMarks that do not contain any references to the database will become part of the variable layer.

Caution: The general SmartMarks concept supports several different overprint modes. However, only Opaque and PostScript overprint modes can be used for variable data printing. The other overprint modes may produce unexpectable results. This is due to the technology which designates that the actual merging is done on the press machine, which currently does not support other overprint modes.

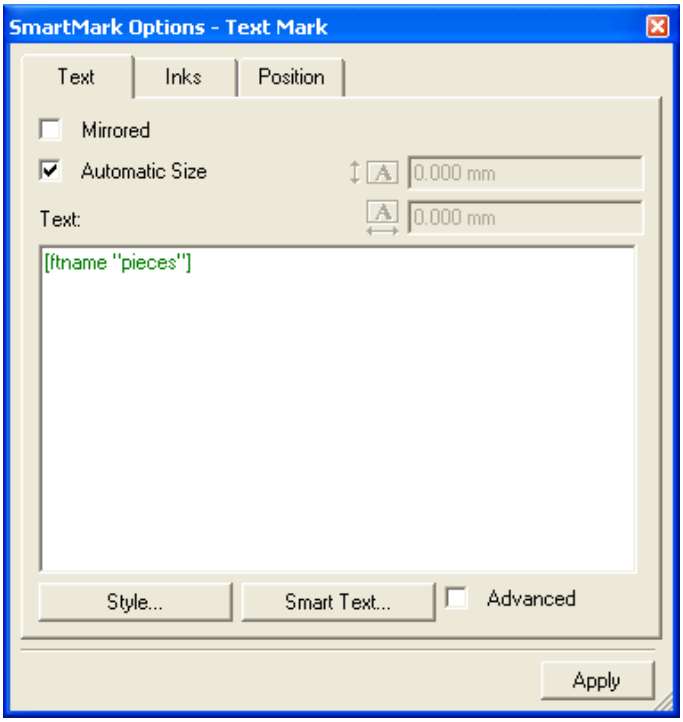
3.3.1 Inserting Text Marks

Use File SmartNames in combination with Text Marks to define variable sections of the text.

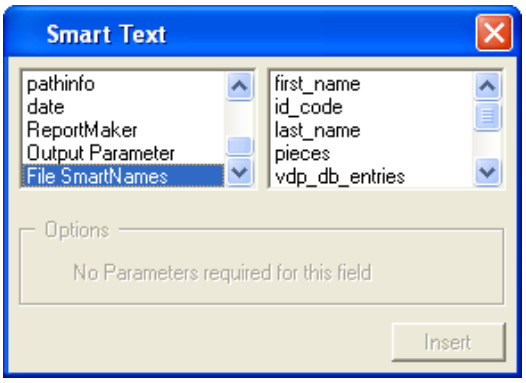
To insert a Text Mark with variable text:

1. Create a new Text Mark.

   The SmartMark Options - Text Mark dialog box appears.



2. Click the 'Smart Text…' button to easily enter a File SmartName.
3. Select File SmartNames from the list on the left of the Smart Text dialog box.



4. Select the appropriate field from the list of field names from the database.
5. Click Insert to insert the File SmartName into the Text Mark.

Example

If the database has a column named 'description' then the SmartName reference inside the Text Mark will be:

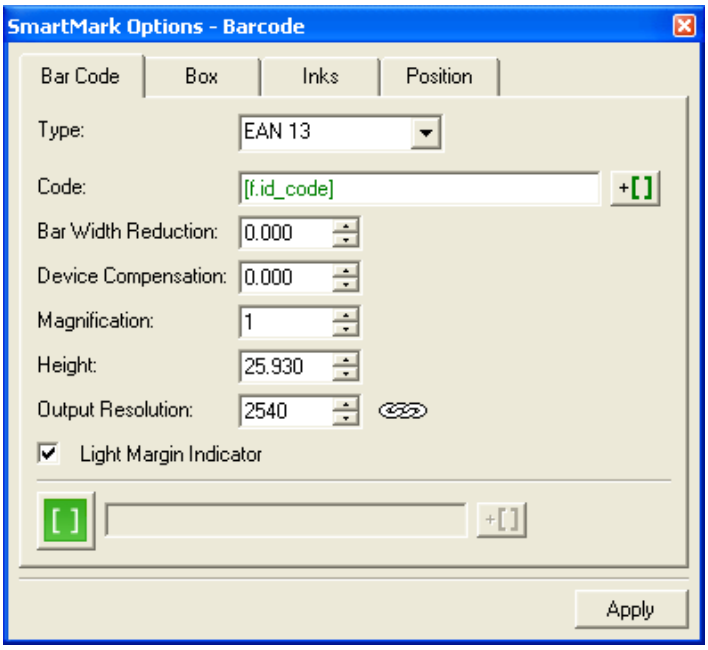
```
Name: [f.tname 'firstname']
```

3.3.2 Inserting Barcode Marks

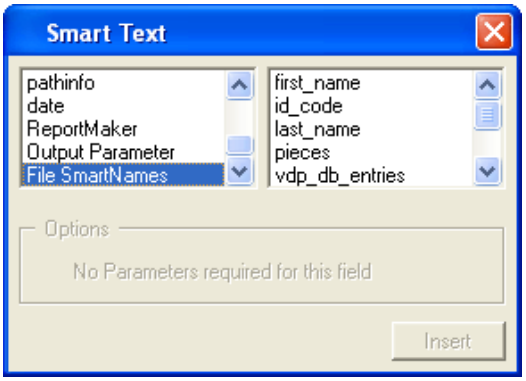
Use File SmartNames to define variable sections of the bar code value. In most cases the whole value will be taken from database.

To insert a Barcode Mark with variable text:

1. Create a new Barcode Mark.
   The SmartMark Options - Barcode dialog box appears.



2. Click the Smart Text button to easily enter a File SmartName.
   The Smart Text dialog box appears.
3. Select one of the items and click 'OK' to insert the Smart Text into the Barcode Mark.



The bar code types provided here are equal to the PackEdge bar code types, so for the properties of each bar code please consult the Esko PackEdge documentation.

It is possible to modify File SmartNames to see how the variable part will look like with different input. All changes can be done in Production menu > File SmartNames.

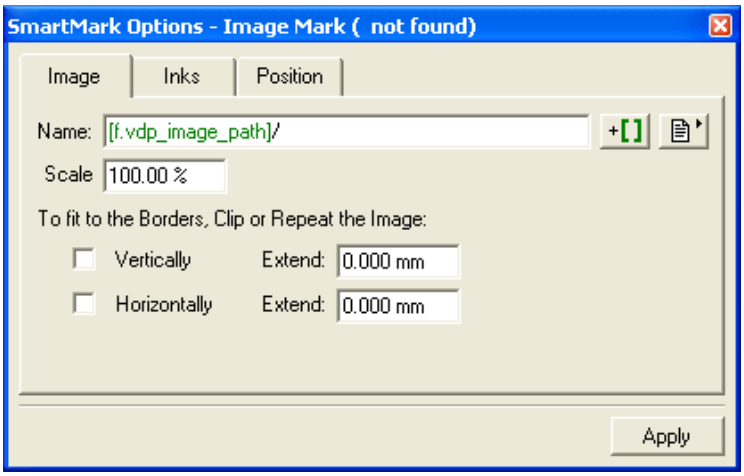
To update a SmartMark with a new value, open the corresponding SmartMark dialog box and press 'Apply'. The new value will replace the old one.

3.3.3 Inserting Image Marks

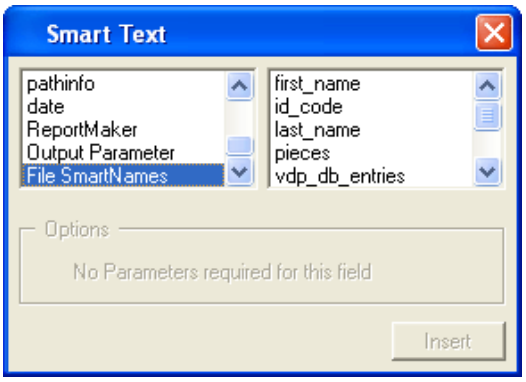
Use File SmartNames to define variable images in your design.

In the Image tab you can define the image location. Insert a File SmartName using

1. Create a new Image Mark.
   The SmartMark Options - Image Mark dialog box appears.



2. Click the Smart Text button (+[]) to define a variable section of the image location.
   The Smart Text dialog box appears.



3. Select File SmartNames and choose the appropriate item from the list on the right.
4. Click 'Insert' to insert the File SmartName into the Name field.

The `[f.vdp_image_path]` SmartName, which can be defined using the File SmartNames dialog box is already prefilled for you. This SmartName represents a path to the folder where all the Image Marks data are located. When you specify this folder in the File SmartNames dialog box, you do not have to specify the full path to the image in the database fields.

Example

The full path to an image looks like this:

```
\\server\share\customer\job\images\image001.ct
```

Use 'Image Path' to specify the fixed part of the path:

```
\\server\share\customer\job\images\
```

Then in the ImageMark settings you can specify the full path as follows:

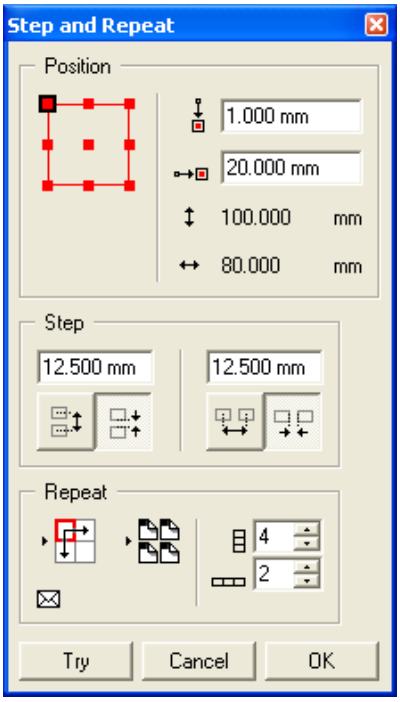
```
[f.vdp_image_path]\image[vdp_index].ct
```

It is recommended to use the Image Path feature, since it makes your job more portable. If the folder with the images is moved, the job can easily be adapted without having to modify any other ImageMark settings.

3.4 Step and Repeat in Plato

The repetition for variable data printing is created in Esko Plato. The result is saved as a PDFPLA file.

The only supported format is PDFPLA. STA and PDFSTA files are not supported. According to Esko standards, all jobs are assumed to be portrait-oriented. Only stations contained in grids are recognized by the expansion task.



- Select 'Step and Repeat' or 'QuickStep repetition chart...' from the Tools menu to create a grid.
  A grid created via 'Filled Step and Repeat' is not supported in a multiple-grid repetition.
- When setting options in the 'Step and Repeat' dialog box, make sure that first cell of the repetition is placed in the top left corner, otherwise the numbering will be incorrect.
- All the pages generated by the expansion task will have the same background. Therefore, 'head turn' repetition is not supported. All pages have to start with the 'head up', so the result will not be correct for pages with an odd number of rows.
- SmartMarks generated in Esko Plato will be placed into the fixed part of a job. It is highly recommended to use the capability of a SmartMark named 'Grid Mark'. Grid Marks can attach

graphic symbols with respect to the repetition layout. This can be useful when you have to create die cut marks. Note that a Grid Mark is supported for a single grid repetition only.

3.5 Launch the Server Tasks

BackStage offers two tasks that process the variable data: Expand Variable Data to PPML File and RIP to WS4000 Series.

Note:

Both the Expand Variable Data to PPML File and RIP to WS4000 Series tasks can be used in chain and can thus be executed as a workflow.

1. Launch the Expand Variable Data to PPML File task in the BackStage Pilot.

   The expansion task will generate fixed and variable parts of a job, based on a design file, a repetition file and a database. All information required is stored in a PDFPLA file, which is the input file for the task. Please refer to the chapter on 'Server Tasks' for more information.

   The output of the task is one PDF file with a fixed background and several PDF files, each of them containing the variable part of a specific sheet. The output files are linked together by a PPML file. This file describes how the fixed and the variable parts should be combined.

   It is recommended to place the output files in a temporary folder of the job. The specified path in the task panel may look like this: [joburl]/tmp_expansion/[date]_[time]
2. Once the Expand Variable Data to PPML File task finishes, start the RIP To WS4000 Series task to launch Esko FlexProof.

   The input file for the RIP To WS4000 Series task is the PPML file.

If the RIP To WS4000 Series task has finished successfully, the input files from the expansion task can be deleted. Please note that the files are not deleted automatically.

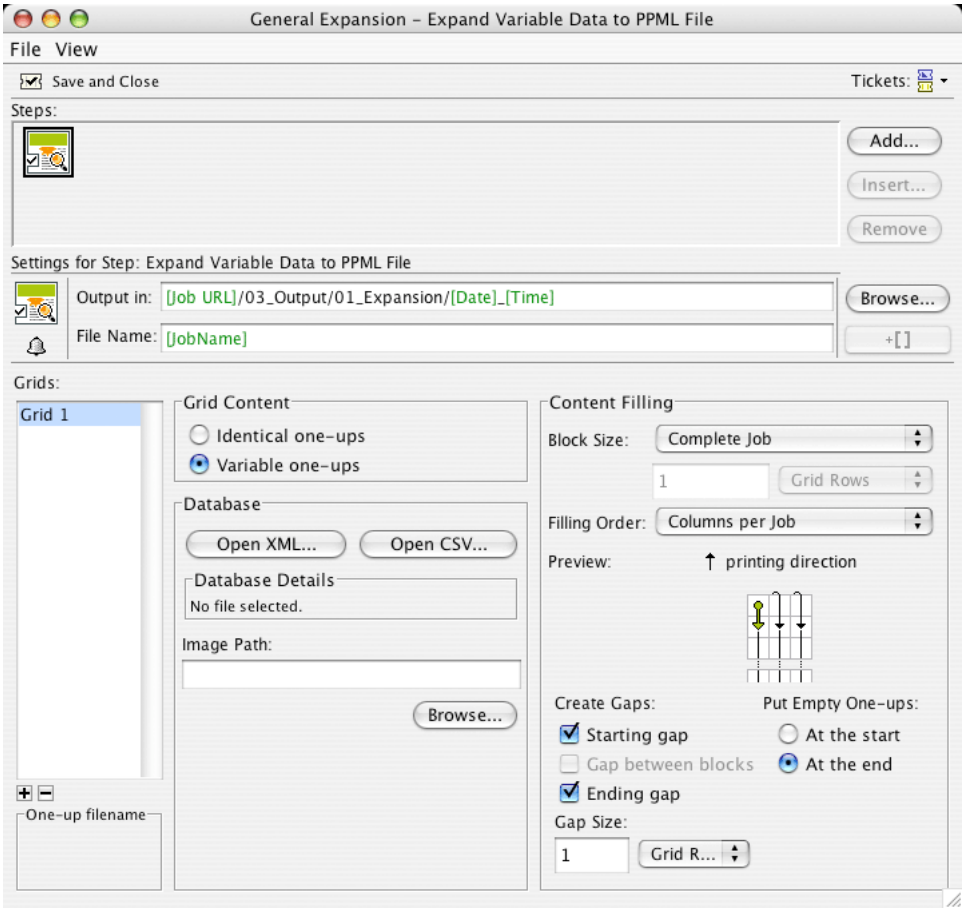
4. BackStage Server Tasks

The BackStage Server offers two tasks to handle variable data in a PackEdge workflow: Expand Variable Data to PPML File, and RIP To WS4000 Series.

The Expand Variable Data to PPML File, and RIP To WS4000 Series can be chained into a workflow ticket.

4.1 Expand Variable Data to PPML File: Configuration

The Expand Variable Data to PPML File task ticket contains settings for defining grids, grid contents, the database and content filling.



4.1.1 Grids

For variable data, multiple grids are supported.

The VDP parameters must be specified for each grid individually.To specify the parameters for a grid, select the grid from the Grids list.

The options for each grid are stored individually, so a change of parameters for one grid will never influence the output of another grid on a frame.

4.1.2 Grid Contents

The expansion task can handle two types of grid contents.

- '**Identical One-ups**' does not include any variable elements. It will fill the empty space on a substrate that is unused by variable data.
- '**Variable One-ups**' carries the design with variable data elements. The main difference for these two types of contents is the absence of a database specification and ordering pattern possibilities for the 'Identical One-ups' option.

Following enumeration gives you an overview of the differences:

Identical one-ups:

- Total number of one-ups for repetition
- Block size (optional)
- Gap size
- Gap at the beginning of the job (optional)
- Gap at the end of the job (optional)

Variable one-ups

- XML database – file URL
- CSV database – one or two file URLs + other options (delimiter character, …)
- Image Path URL – when Image Marks are used (optional)
- Block size: job, sheet or custom
- Gap at the beginning of the job (optional)
- Gap between blocks (optional)
- Gap at the end of the job (optional)
- Gap size
- Ordering
- Empty one-ups

4.1.3 Database

This parameter defines the database which will be used for expansion. The database file can be different from the one used for loading File SmartNames.The only requirement is that the identical

column names are used. The database can be either in XML or a CSV file format and must use the UTF-8 encoding.

XML Database

This type of database is a file in the eXtensible Markup Language format. Clicking 'Open from XML...' opens a file browser, which allows you to select the XML file. Please note that the XML file must be stored within a container, in order to be visible to the BackStage server which performs the expansion task.

CSV Database

A CSV is a Comma Separated Value file. The 'Open from CSV...' button opens a dialog box, where the operator has to specify a URL to the database file, its Encoding and Field Separator options.If field names are not stored in the first line of the database file, you can specify a separate file with the required information by using the Field Names part of the dialog box.

CSV file format is the preferred format, since the expansion performance for larger databases is higher. An overview of all the settings will be displayed in the Variable Data page of the stepX dialog.

Image Path

The location specified here is used to fill in the `vdp_image_path`' SmartName, which might be necessary if the job uses Image Marks. It must contain a valid path to the images folder.

Please note, that the value specified in the File SmartNames dialog is used for designing / preview purposes only and therefore is not used by the expansion task. Therefore it needs to be specified once again here as an expansion task parameter.

4.1.4 Content Filling

This part of the dialog box is used to specify how the database values will flow through the variable job. Please note, that according to Variable Data Printing standards, all the jobs are **assumed to be portrait-oriented**.

Block Size

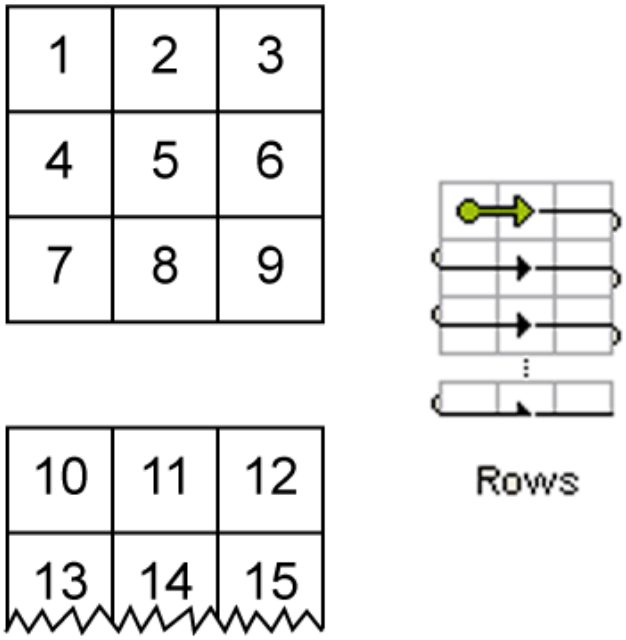
Following options are available:

- Complete Job: This option defines that database will flow through the job without any interruptions.
- One Frame: The size of a frame is equal to size of the document defined in stepX.
- Custom: The size of a block can be user-defined. Units that can be used are grid rows, frames, meters or feet. For units such as meters or feet, rounding to grid rows is applied.

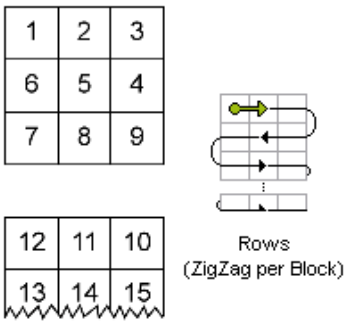
Filling Order

Following options are available:

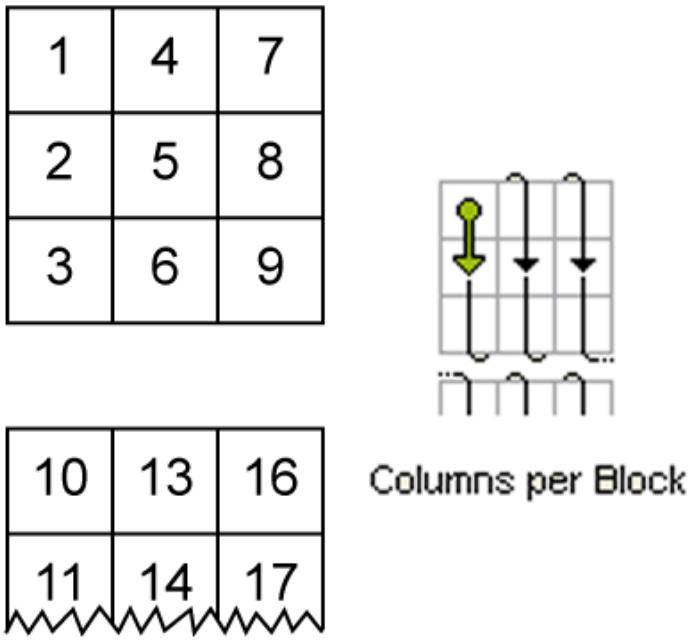
- **Rows:** The numbering is restarted when the end of a row is reached and continues from the left cell of the next row. The starting position is the top left cell of each block.



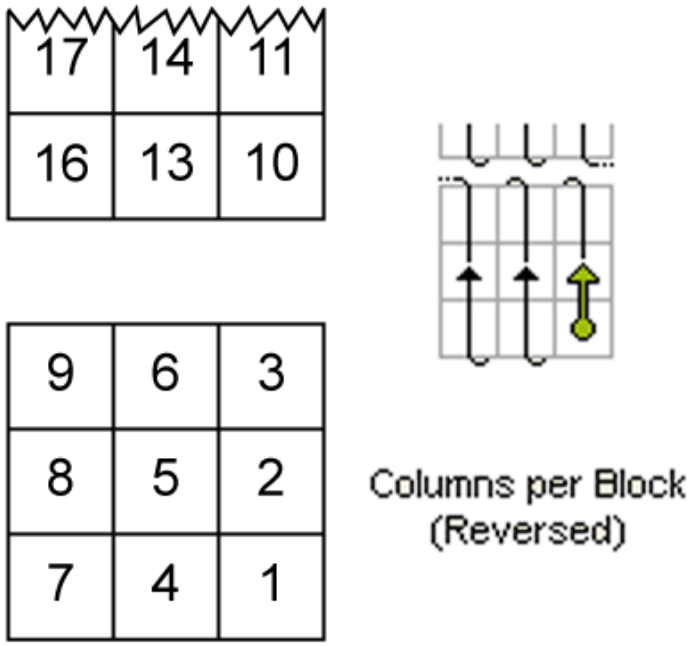
- **Rows (ZigZag):** The numbering pattern alternates direction for each row and is restarted once the beginning of a row is encountered. The starting position is the top left cell of each block.



- **Columns per Block:** The numbering pattern is restarted when the end of a column in a block is reached and continues from the top cell of the next column. The starting position is the top left cell of each block.Note: If the Block Size is set to ‘Complete Job’, then this filling order is equal to 'Columns per Job'.

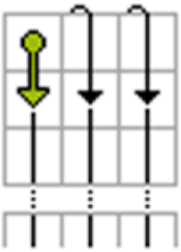


- **Columns per Block (Reversed):** The numbering is restarted when the end of a column in a block is reached and continues from the bottom cell of the next column. The starting position is the bottom right cell of each block. Note: If the Block Size is set to ‘Complete Job’, then this filling order is equal to 'Columns per Job (Reversed)'.



- **Columns per Job:** The numbering is restarted when the last cell of a column in the job is reached and continues from the top cell of the next column. The starting position is the top left cell of the job.

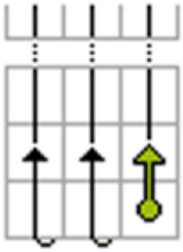
1	7	13
2	8	14
3	9	15
4	10	16
5	11	17



Columns per Job

- **Columns per Job (Reversed):** The numbering is restarted when the last cell of a column in the job is reached and continues from the bottom cell of the next column. The starting position is the bottom right cell of the job.

17	11	5
16	10	4
15	9	3
14	8	2
13	7	1



Columns per Job (Reversed)

Create Gaps

Some jobs need to be interrupted after a certain amount of labels. This amount can be defined using the Block Size parameter. Gaps can be inserted between these blocks as well as to the beginning and to the end of a job.

No one-ups are printed in the gap area, so the press operator can easily identify the gap as a white space. However, all the elements outside the one-up area, such as die cut marks, are printed.

Here you can specify where you want to the gaps to be generated. The following options are available:

- Starting gap This gap will be printed first (with respect to the printing direction), before any variable data are printed.
- Gap between blocksThese gaps will be generated in between Blocks.Note: When Block Size is set to 'Complete job', this toggle is useless.
- Ending gapThis gap will be printed last (with respect to the printing direction), after all the variable data have been printed.

The 'Gap Size' option specifies the size of the gaps in grid rows, frames, meters or feet.

Put Empty one-ups

In general, the database size is not an exact multiple of the number of cells in a block. most cases, one of the blocks cannot be filled completely with database entries. It will be padded with empty one-ups. It will be padded with empty one ups.The toggles **At the start** or **At the end** specify where to put the empty one-ups from Filling Order point of view.

Examples

Printing direction

49	50	51	52			53	54	55	56			57	58	59	60			61	62		
33	34	35	36			37	38	39	40			41	42	43	44			45	46	47	48
17	18	19	20			21	22	23	24			25	26	27	28			29	30	31	32
1	2	3	4			5	6	7	8			9	10	11	12			13	14	15	16

This job was created using the following settings: Block size: 4 grid rows; Filling order: Columns per Job; Gap between blocks: 2 grid rows; Empty One-ups: At the end

17	18	19	20			37	38	39	40			57	58	59	60				
13	14	15	16			33	34	35	36			53	54	55	56				
9	10	11	12			29	30	31	32			49	50	51	52				
5	6	7	8			25	26	27	28			45	46	47	48				
1	2	3	4			21	22	23	24			41	42	43	44			61	62

This job was created using the following settings: Block size: 4 grid rows; Filling order: Columns per Block; Gap between blocks: 2 grid rows; Empty One-ups: At the end

62	61			44	43	42	41			24	23	22	21			4	3	2	1
				48	47	46	45			28	27	26	25			8	7	6	5
				52	51	50	49			32	31	30	29			12	11	10	9
				56	55	54	53			36	35	34	33			16	15	14	13
				60	59	58	57			40	39	38	37			20	19	18	17

This job was created using the following settings: Block size: 4 grid rows; Filling order: Columns per Block (Reversed); Gap between blocks: 2 grid rows; Empty One-ups: At the beginning

4.2 Rip to WS4000 Series: Configuration

When the expansion task finishes, the 'RIP to WS4000 Series' task can be started to launch FlexRip. There are only two minor differences in the ticket dialog compared to a generic rip or proof task.

Note:

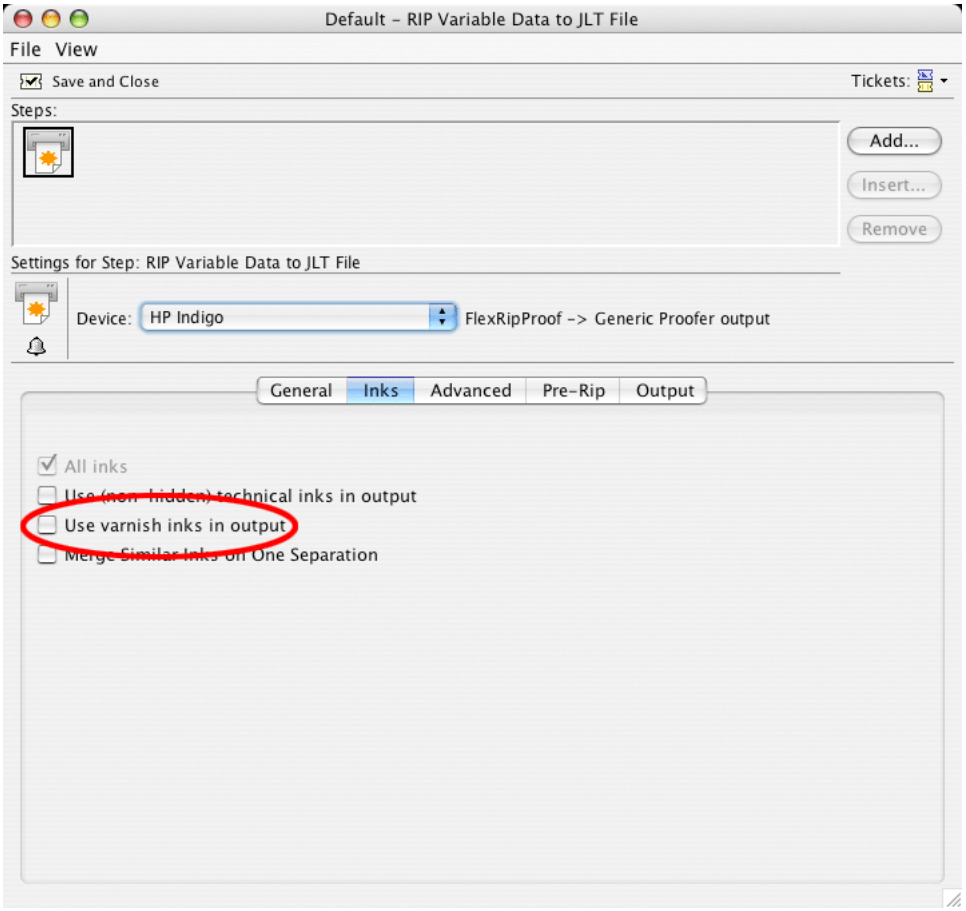
For a description on all the options of the RIP task, please refer to the Esko BackStage documentation.

If the RIP to WS4000 Series task finishes successfully, the temporary files from the expansion task can be deleted. These files are not deleted automatically.

4.2.1 Inks Tab

The Inks tab of the Rip to WS4000 Series task ticket contains a variable data printing-specific option, **Use varnish inks in output**.

The **Use varnish inks in output** option enables or disables interpretation of varnish-type inks by the RIP.



4.2.2 Output Tab

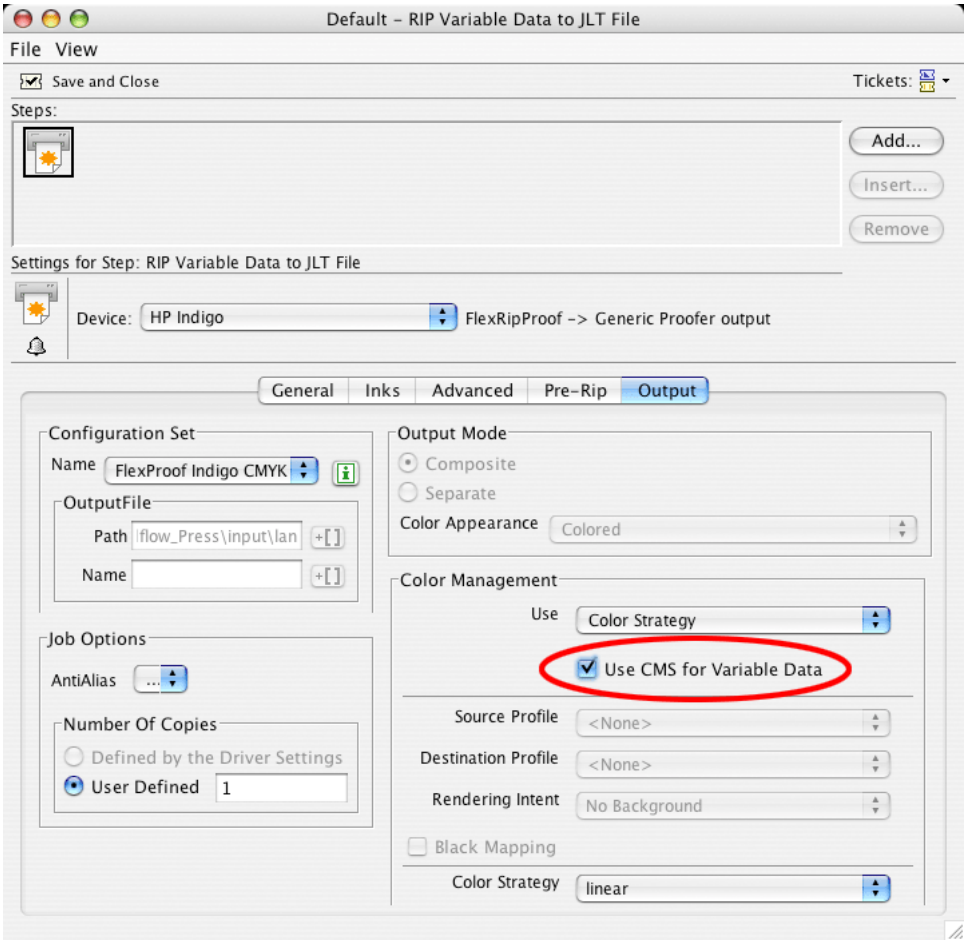
The Output tab of the Rip to WS4000 Series task contains the variable data printing-specific option **Use CMS for Variable Data**.

The **Use CMS for Variable Data** option enables the use of color management for the variable data elements in a design, instead of only color managing the fixed background.

If the variable data are put in a Pantone or a designer ink and the option is turned off, the inks will be converted by using the color strategy 'linear'.

Note:

Turning on color management for variable data slows down the performance significantly.



4.2.3 Known Issues in the RIP

The following are known issues when ripping variable data printing:

Only PostScript and Opaque overprint modes

The overprint modes of the variable part on the fixed part are limited to PostScript overprint and Opaque. This is caused by the fact that the 2 parts are combined on the press, and not on the system. However inside the fixed part of the job all overprint or transparency settings are supported.

Unsupported transparacies in variable layer

Using unsupported transparencies in the variable layer may result in parts of the variable objects not being printed.

No JLT viewer available

Currently no JLT viewer is available. The only way to check ripped data is to view them at HP Indigo press, using the Job Editor.

Files not deleted automatically

Files stored in the 'images' folder at the press PC are not deleted automatically. Make sure that the total number of files located in this folder does exceed 3 000, otherwise operations within this folder may become slower.

CMS for variable data reduces performance

When CMS is involved for the variable part of job, performance is reduced significantly.

No Magenta1 layer for variable data

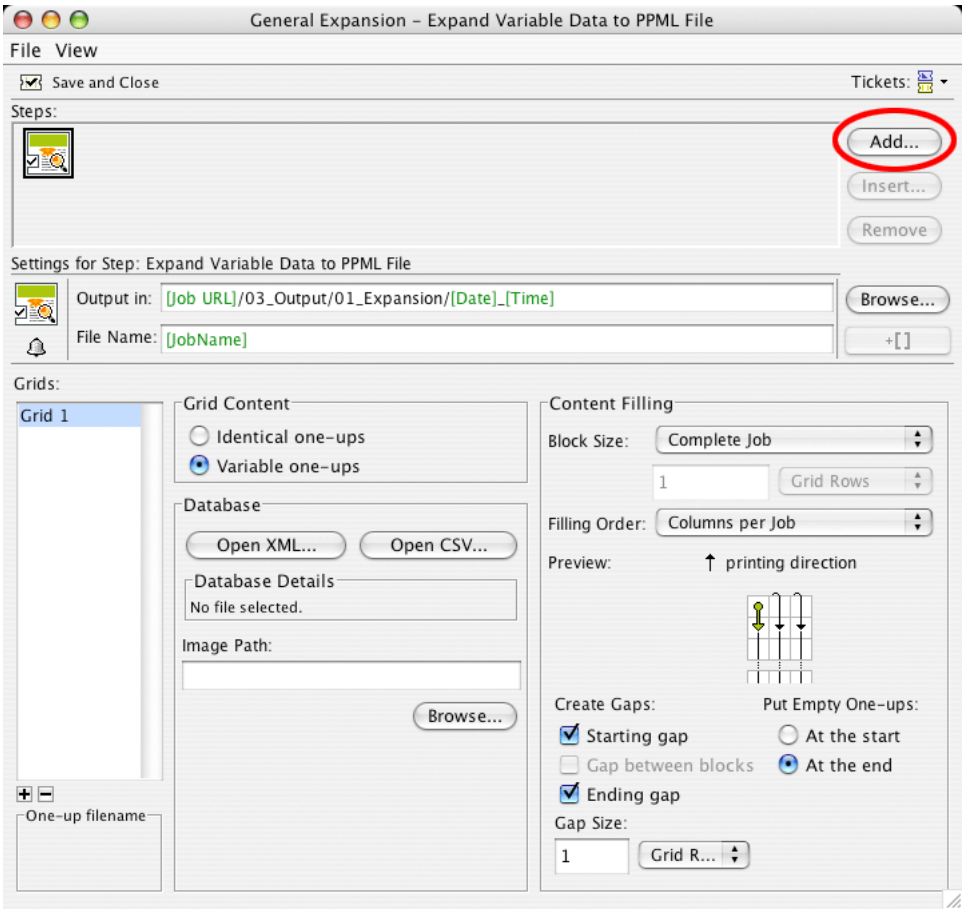
A Magenta1 separation (M1) is generated when Magenta overlaps with an Orange, Violet or Green separation. Using M1, however, is not supported for the variable layer. M1 will never be generated for the variable layer.

4.3 Chaining the Tasks

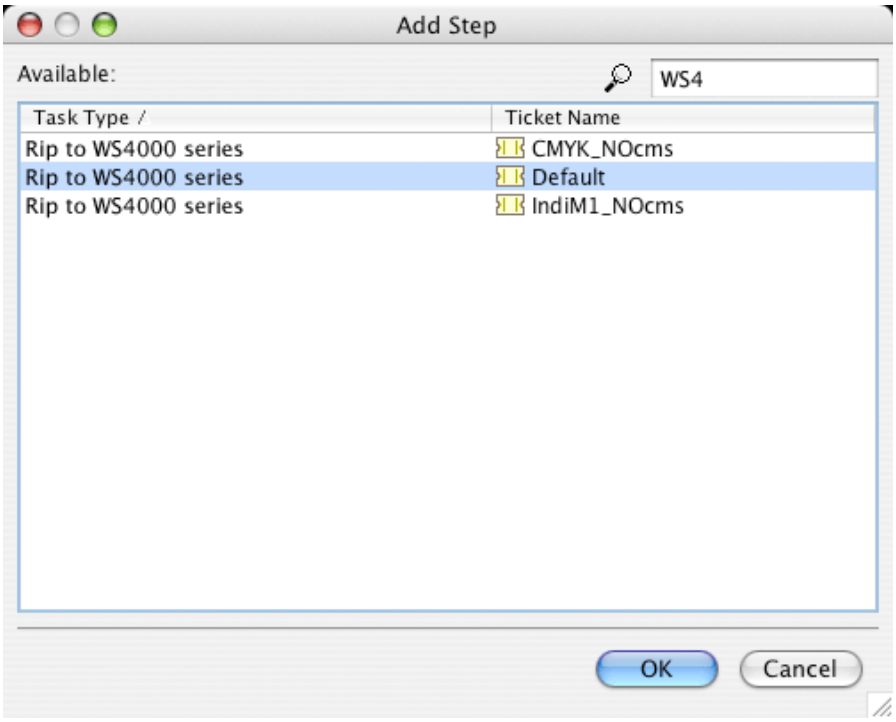
Two or more tickets can be connected in a chain. Consult the Esko BackStage 7.0 User Manual for more detailed instructions.

To chain the variable data printing tickets:

1. Open the Expand Variable Data to PPML File task ticket.



2. Click the Add… button.
   The Add Step dialog box appears.
3. Select the Rip to WS4000 Series task ticket from the Task Type list in the Add Step dialog box.

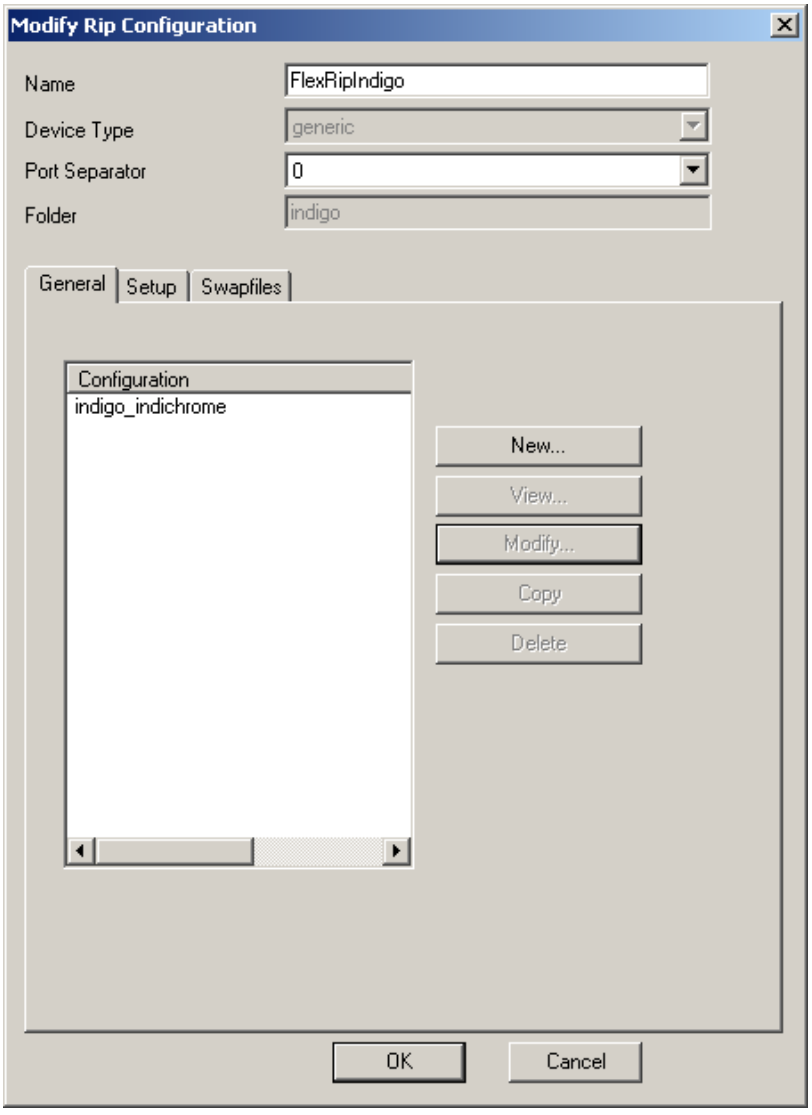


4. Click OK.
   The chain has been created and you can now save the workflow ticket.

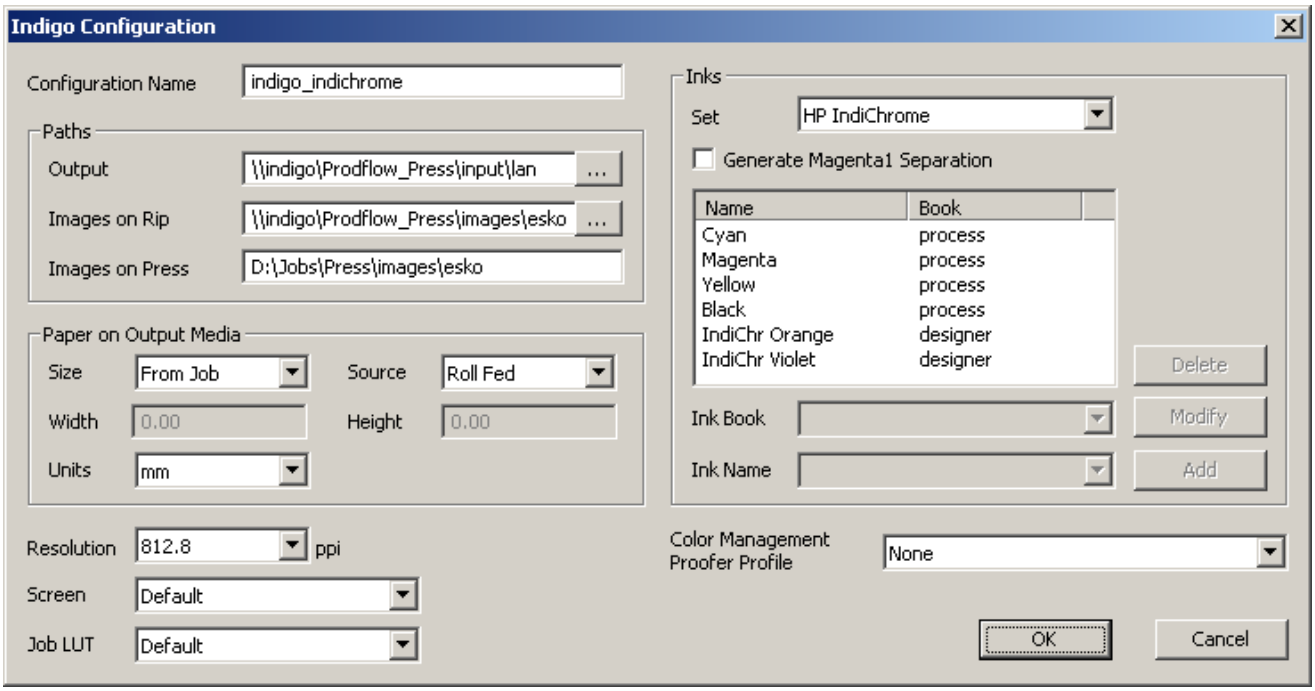
5. Configuration

5.1 FlexRip/Indigo Configuration

To configure the RIP, a FlexRip/Indigo Dispatcher must exist. Usually a default one is created during the installation process. To modify its settings, use the FlexRip Configurator application.



5.2 Dispatcher Configuration



The **Paths** group contains the location of the Master JLT file and of all the variable JLT files (images). The output folder of the RIP is the hotfolder on the press. Please note, that all of the paths must use the UNC format, with the 'Images on Press' field being the only exception, since this path is used by the press to access the variable JLT files stored at its local drive. If your network drive is mapped to a drive letter, an automatic conversion to UNC format will be offered.

Typically paths look similar to this:

Output:

```
\\indigo\Prodflow_Press\input\lan
```

Images on Rip:

```
\\indigo\Prodflow_Press\images\esko
```

Images on Press:

```
D:\Jobs\Press\images\esko
```

5.3 Access to the Press PC

The BackStage server user BGSYSTEM must have write access to the hotfolder of the press PC. If not, the 'RIP to WS4000 Series' task will cause an error when writing the Master JLT to the hotfolder.

The configuration of access rights depends on the OS where the BackStage server runs:

- **Windows 2000 Server**: make sure that you have access to the Administrator account of the press PC. Create the user BGSYSTEM on the press PC with the same password as the one on the BackStage server system and grant this user access to the required folders.
- **Windows 2003 Server:**
  - Log in as user BGSYSTEM on the BackStage server.
  - Map the hotfolder of the Press:

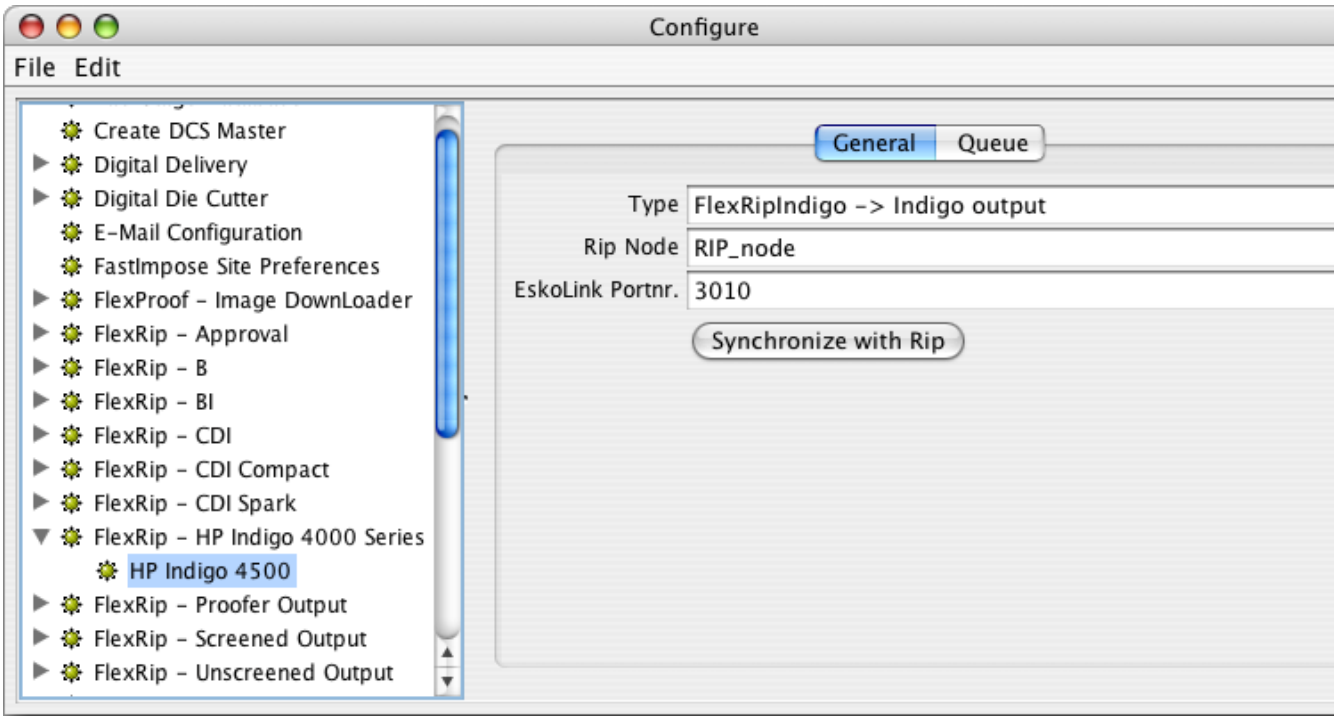
```
net use \\indigo\Prodflow_Press
/SAVECRED
user: . . .
password: . . .
```

`/SAVECRED` stores the credentials permanently. During subsequent logons the share on the press will be automatically accessible.

5.4 Dispatcher connection with the BackStage server

The BackStage server must be configured in order to be able to send data to the HP Indigo Dispatcher. A new proofer device that is linked to the HP Indigo Dispatcher must be created. This can be done using the Configure application (BackStage Pilot > Tools menu > Configure).

In the left pane of the Configure application, select 'FlexRip – HP Indigo 4000 Series' and click File > New to create a new device. In the right pane, enter the port number of the Indigo Dispatcher and click 'Synchronize with Rip'.


Configure
File Edit
Create DCS Master
Digital Delivery
Digital Die Cutter
E-Mail Configuration
FastImpose Site Preferences
FlexProof - Image DownLoader
FlexRip - Approval
FlexRip - B
FlexRip - BI
FlexRip - CDI
FlexRip - CDI Compact
FlexRip - CDI Spark
FlexRip - HP Indigo 4000 Series
HP Indigo 4500
FlexRip - Proofer Output
FlexRip - Screened Output
FlexRip - Unscreened Output
General
Queue
Type
FlexRipIndigo -> Indigo output
Rip Node
RIP_node
EskoLink Portnr.
3010
Synchronize with Rip